This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1. (Previously presented) A method of implementing a DCT in a GPU, comprising:

separating an image into blocks of pixels;

processing each block of pixels, in parallel, within at least one shader module, the processing comprising:

multiplying a column or row of pixels with a predetermined matrix to generate a corresponding set of output pixels;

determining sets of scanlines based on the sets of output pixels; and

for each set of scanlines, sampling at least a portion of the pixels comprised within the scanlines and pixels relative to the scanlines, and multiplying the sampled pixels with a row or column of the predetermined matrix,

wherein said multiplying a column or row of pixels with a predetermined matrix to generate a corresponding set of output pixels, determining, and sampling the pixels are performed by said at least one shader module.

2. (Previously presented) The method of claim 1, wherein the multiplying a column or row of pixels with a predetermined matrix to generate a corresponding set of output pixels, determining, and sampling the pixels are performed in the GPU.

3. (Original) The method of claim 1, wherein each corresponding set of output pixels corresponds to a textured line across the pixels in the blocks of pixels.

4. (Original) The method of claim 1, wherein sampling the pixels comprised within the scanlines comprises using a separate shader for each set of scanlines.

5. (Original) The method of claim 4, further comprising defining an array of coordinate offsets to neighboring pixels, wherein the shader accesses the pixels in the scanlines using the offset array.

6. (Original) The method of claim 4, wherein the same shader can be used for each pixel in a scanline.

7. (Previously presented) A method of processing pixels, comprising;

separating an image into blocks of pixels;

processing each block within at least one shader module, the processing comprising:

creating a polyline of pixels for each column or row in each block of pixels; and

creating a line for each row or column in each block of pixels, wherein the rows or columns correspond to the polylines created for each column or row; and

wherein said creating a polyline and creating a line are performed by said at least one shader module.

8. (Original) The method of claim 7, further comprising:

Creating a polyline of pixels for each row or column in each block of pixels; and

Creating a line for each column or row in each block of pixels, wherein the rows or columns correspond to the polylines created for each row or column.

9. (Original) The method of claim 7, further comprising:

determining sets of scanlines based on the lines created for each row or column in each block of pixels; and

for each set of scanlines, sampling the pixels comprised within the scanlines and multiplying the sampled pixels with a row or column of a predetermined matrix.

10. (Original) The method of claim 7, wherein the steps of creating are performed in a graphics processing unit (GPU).

11. (Previously presented) A method of processing pixels, comprising:

separating an image into blocks of pixels;

processing each block within at least one shader module, the processing comprising:

determining a polyline of pixels for each column or row in each block of pixels;

for each pixel in the polyline,

sampling at least a portion of the other pixels in the corresponding column or row that lies along the polyline and pixels relative to the column or row;

multiplying each of the other pixels by a DCT coefficient from a predetermined matrix to generate resultant values; and

adding the resultant values together to generate a resulting value,

wherein said multiplying and adding are performed by said at least one shader module.

12. (Original) The method of claim 11, further comprising biasing and scaling at least one of the polyline of pixels, the resultant values, and each resulting value for each pixel.

13. (Previously presented) A method of processing pixels comprising:

separating an image into blocks of pixels;

processing each block within at least one shader module, the processing comprising:

for each column in a block of pixels, setting up a shader and rendering a scanline; and

for each row in a block of pixels, setting up a shader and rendering a column; and

wherein the setting up and the rendering are performed by said at least one shader module.

14. (Previously presented) The method of claim 13, wherein setting up the shaders and the rendering are performed in the GPU.

15. (Previously presented) A system to program a GPU to implement a DCT, comprising:

adapting a processing unit to receive blocks of pixels into which an image has been separated, and processing each block of pixels, in parallel, within at least one shader module, the processing comprising:

multiplying a column or row of pixels of an image with a predetermined matrix to generate a corresponding set of output pixels;

determining sets of scanlines based on the sets of output pixels; and

for each set of scanlines, sampling the pixels comprised within the scanlines and multiplying the sampled pixels with a row or column of the predetermined matrix and

wherein said setting up and the rendering are performed by said at least one shader module.

16. (Canceled)

17 (Previously presented) The system of claim 15, further comprising a CPU coupled to the GPU by a system bus, the CPU capable of separating the image into the blocks of pixels.

18. (Previously presented) The system of claim 15, wherein each corresponding set of output pixels corresponds to a textured line across the pixels in the blocks of pixels.

19. (Previously presented) The system of claim 15, wherein the GPU comprises a separate shader for sampling the pixels comprised within each set of the scanlines.

20. (Original) The system of claim 19, wherein the GPU defines an array of coordinate offsets to neighboring pixels, wherein the shader accesses the pixels in the scanlines using the offset array.

21. (Original) The system of claim 19, wherein the same shader can be used for each pixel in a scanline.

22. (New) A method of implementing low-level video processing in a GPU, comprising:

separating an image into blocks of pixels;

processing each block of pixels, in parallel, within at least one shader module, the processing comprising:

multiplying a column or row of pixels with a predetermined matrix to generate a corresponding set of output pixels, determining, and sampling the pixels are performed in the GPU;

determining sets of scanlines based on the sets of output pixels; and

for each set of scanlines, sampling at least a portion of the pixels comprised within the scanlines and pixels relative to the scanlines, and multiplying the sampled pixels with a row or column of the predetermined matrix using a separate shader for each set of scanlines;

defining an array of coordinate offsets to neighboring pixels, wherein the shader accesses the pixels in the scanlines using the offset array; and

wherein said multiplying a column or row of pixels with a predetermined matrix to generate a corresponding set of output pixels, determining, and sampling the pixels are performed by said at least one shader module.